BAB II

KAJIAN TEORI

Pembahasan pada bagian ini akan menjadi dasar teori yang akan digunakan untuk membahas bab berikutnya. Dasar teori yang akan dibahas pada bab ini adalah optimisasi, fungsi, pemrograman linear, pemrograman nonlinear, *separable programming* dan algoritma genetika.

A. Optimisasi

Optimisasi berasal dari bahasa inggris *optimization* yang memiliki arti memaksimalkan atau meminimalkan sebuah fungsi yang diberikan untuk beberapa macam kendala (Licker, M. D, 2003 : 170). Secara lebih sederhana dapat dijelaskan bahwa optimisasi adalah suatu cabang dari ilmu matematika yang digunakan untuk memaksimumkan atau meminimumkan fungsi tujuan dengan mempertimbangkan beberapa kendala yang diberikan.

Konsep optimisasi mempunyai peranan penting dalam menyelesaikan suatu masalah yang ada dalam kehidupan sehari-hari. Prosedur pemecahan masalah optimisasi yang biasa digunakan yaitu memodelkan persoalannya ke dalam sebuah program matematis dan kemudian memecahkannya dengan menggunakan teknik-teknik atau metode optimisasi seperti program linier, program nonlinier, program tujuan ganda, dan metode-metode lainnya yang sudah berkembang saat ini.

B. Analisis Regresi Kuadratik

Pada prinsipnya analisis regresi merupakan pencarian kurva yang mewakili hubungan satu set data. Data tersebut dapat berupa data yang diperoleh berdasarkan hasil pengamatan, percobaan atau data statistik. Sedangkan regresi kuadratik adalah salah satu metode regresi yang digunakan untuk menentukan fungsi polynomial derajat dua yang paling sesuai dengan kumpulan titik data (x_n, y_n) dengan banyaknya titik data ≥ 3.

Secara umum persamaan regresi kuadratik adalah (Djoko Luknanto, 1992 : 8)

$$Y = a + bX + cX^2$$

dengan

$$\sum Y_i = na + b\sum X_i + c\sum X_i^2$$

$$\sum X_i Y_i = a\sum X_i + b\sum X_i^2 + c\sum X_i^3$$

$$\sum X_i^2 Y_i = a\sum X_i^2 + b\sum X_i^3 + c\sum X_i^4$$

dengan a, b, c adalah konstanta, x adalah variabel bebas (*independent*) dan y adalah variabel terikat (*dependant*) dan n adalah banyaknya data.

Contoh 2.1

Tentukan persamaan regresi kuadratik berdasarkan pasangan titik

{(1,-1) ,(2,2),(3,7),(4,14)}.

Berdasarkan data di atas diperoleh perhitungan sebagai berikut :

Tabel 2.1 Perhitungan berdasarkan Contoh 2.1

x	y	x^2	x^3	x^4	xy	x^2y
1	-1	1	1	1	-1	-1
2	2	4	8	16	4	8
3	7	9	27	81	21	63
4	14	16	64	256	56	224
$\sum x = 10$	$\sum y = 22$	$\sum x^2 = 30$	$\sum x^3 = 100$	$\sum x^2 = 354$	$\sum xy = 80$	$\sum x^2y = 294$

Dari tabel diatas diperoleh 3 persamaan linear berdasarkan rumus regresi kuadratik yaitu

$$22 = 4a + 10b + 30c$$

$$80 = 10a + 30b + 100c$$

$$294 = 30a + 100b + 354c$$

Selanjutnya dengan eliminasi dan substitusi diperoleh nilai $a = -2, b = 0, c = 1$ sehingga didapatkan persamaan regresi

$$y = x^2 - 2$$

Selanjutnya proses penghitungan regresi kuadratik dalam skripsi ini akan menggunakan bantuan software geogebra dengan perintah *command fitpoly*. *Command fitpoly* merupakan suatu perintah dalam program geogebra yang

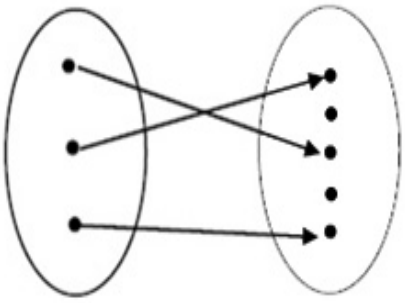
digunakan untuk mencari regresi polynomial berdasarkan data yang diinput dalam spreadsheet.

C. Fungsi

Pembahasan tentang fungsi tidak lepas dari istilah relasi. Relasi adalah aturan yang memasangkan anggota suatu himpunan dengan anggota himpunan yang lain. Sedangkan fungsi atau pemetaan merupakan suatu relasi yang bersifat khusus. Fungsi dari himpunan A ke himpunan B adalah suatu relasi dimana setiap anggota himpunan A dipasangkan dengan tepat satu anggota himpunan B.

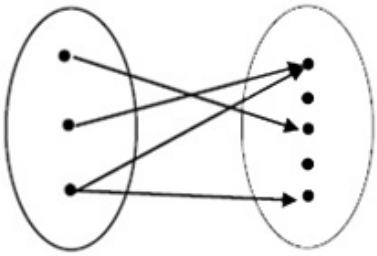
Definisi 2.1. Fungsi (Purcell, E. J. & D. Verberg, 1987 : 57) *Suatu fungsi f adalah suatu aturan padanan yang menghubungkan setiap obyek x dalam suatu himpunan, yang disebut daerah asal, dengan sebuah nilai tunggal f(x) dari suatu himpunan kedua (daerah kawan).*

Gambar 2.1 berikut memberikan ilustrasi untuk suatu fungsi.



Gambar 2. 1 Ilustrasi Fungsi

Gambar 2.2 berikut memberikan ilustrasi bukan fungsi



daerah asal daerah kawan

Gambar 2. 2 Ilustrasi bukan fungsi

Berdasarkan Definisi 2.1 dapat disimpulkan bahwa pada Gambar 2.1, merupakan fungsi karena setiap titik pada daerah asal memiliki satu pasangan pada daerah kawan. Sedangkan Gambar 2.2. bukan fungsi karena ada titik pada daerah asal yang memiliki dua (lebih dari satu) pasangan pada daerah kawan.

Konsep fungsi cembung merupakan hal yang penting dalam permasalahan optimisasi. Sebelum membahas fungsi cembung (konveks) perlu dipahami dulu konsep himpunan cembung.

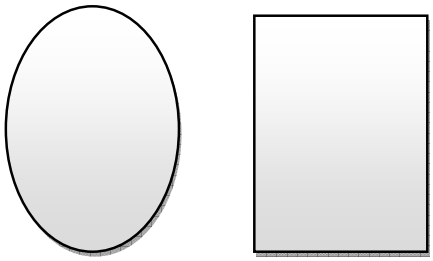
Definisi 2.2 (Bazaraa,2006:40) *Himpunan* $S \subseteq R^n$ *dikatakan cembung jika segmen garis yang menghubungkan dua titik pada himpunan S juga berada di himpunan S. Dengan kata lain, jika* $x_1, x_2 \in S$ *maka* $\lambda x_1 + (1-\lambda)x_2$ *juga anggota* S *untuk* $\lambda \in (0,1)$.

Contoh 2.2

Himpunan bilangan positif $\boldsymbol{P} \subseteq \boldsymbol{R}$, $\forall\, x, y \in \boldsymbol{P}$, maka $z = cx + (1-c)y \in \boldsymbol{P}$, $c \in [0,1]$.

Himpunan titik-titik di ***P*** yang merupakan bilangan positif, dimana sebarang pasangan titik di dalam himpunan ***P*** jika dihubungkan oleh garis maka seluruh titik pada garis tersebut juga terdapat di ***P.***

Gambar 2.3 berikut merupakan ilustrasi himpunan cembung



Gambar 2.3 (a),(b) himpunan cembung

Gambar 2.4 berikut merupakan ilustrasi bukan himpunan cembung

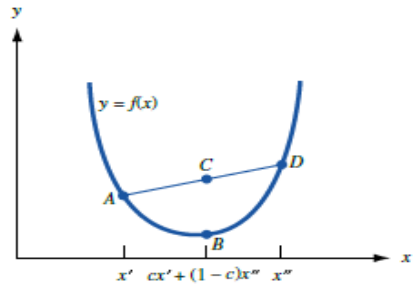
Gambar 2.4 bukan himpunan cembung

Definisi 2.3 Fungsi Cembung (Bradley, 1976:416) *Diketahui* $f: S \rightarrow R$ *dengan S adalah himpunan cembung yang tidak kosong di* R^n. *Fungsi f(x) dikatakan fungsi cembung di S ketika*

$$f(\lambda x_1 + (1-\lambda)x_2) \leq \lambda f(x_1) + (1-\lambda) f(x_2)$$

untuk setiap $x_1, x_2 \in S$ *dan untuk* $0 \leq \lambda \leq 1$.

Contoh fungsi cembung dapat digambarkan sebagai berikut



Gambar 2.5 Contoh Fungsi Cembung

Titik $A = (x', f(x'))$,

Titik $D = (x'', f(x''))$,

Titik $C = (cx' + (1-c)x'',\ cf(x') + (1-c)f(x''))$,

Titik $B = (cx' + (1-c)x'',\ f(cx' + (1-c)x''))$.

Berdasarkan gambar di atas didapatkan bahwa

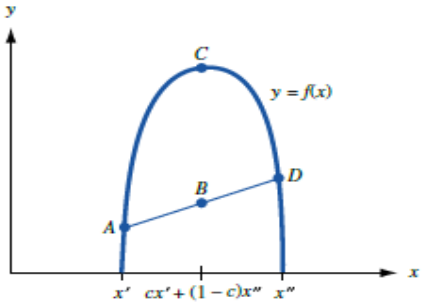
$f(cx' + (1-c)x'') \leq cf(x') + (1-c)f(x'')$

Definisi 2.4 Fungsi Cekung (Bradley, 1976:416) *Diketahui* $f: S \to R$ *dengan S adalah himpunan cembung yang tidak kosong di* R^n. *Fungsi f(x) dikatakan fungsi cekung di S ketika*

$$f(\lambda x_1 + (1-\lambda)x_2) \leq \lambda f(x_1) + (1-\lambda)f(x_2)$$

untuk setiap $x_1, x_2 \in S$ *dan untuk* $0 \leq \lambda \leq 1$.

Contoh fungsi cekung dapat digambarkan sebagai berikut



Gambar 2.6 Contoh Fungsi Cekung

Titik $A = (x', f(x'))$,

Titik $D = (x'', f(x''))$,

Titik $C = (cx' + (1-c)x'',\ f(cx' + (1-c)x"))$,

Titik $B = (cx' + (1-c)x'',\ cf(x') + (1-c)f(x"))$.

Berdasarkan gambar di atas didapatkan bahwa

$f(cx' + (1-c)x") \geq cf(x') + (1-c)f(x")$

Fungsi cembung dan fungsi cekung dapat ditentukan dengan menggunakan turunan kedua. Menurut Hillier & Lieberman (2001), fungsi cekung dan cembung pada suatu fungsi satu variabel dapat ditentukan melalui Teorema 2.1 berikut :

Teorema 2.1. Tes Kecembungan dan Kecekungan Fungsi Satu Variabel (Hillier & Lieberman, 2001 : 1159)

1. *Fungsi f(x) cembung jika dan hanya jika turunan kedua f(x) yaitu* $\frac{d^2f(x)}{dx^2} \geq 0$ *untuk setiap nilai x yang diberikan.*

2. *Fungsi f(x) cekung jika dan hanya jika turunan kedua f(x) yaitu* $\frac{d^2f(x)}{dx^2} \leq 0$ *untuk setiap nilai x yang diberikan.*

Bukti:

Misalkan f konveks pada interval terbuka *I* dan$f: I \to \mathbb{R}$. Untuk tiap $c \in I$, diperoleh

$$f"(c) = \lim_{h \to 0} \frac{f(c+h) - 2f(c) + f(c-h)}{h^2}$$

Selanjutnya pilih h cukup kecil sedemikian hingga $c - h$ dan $c + h$ ada di *I*. Maka $c = \frac{1}{2}[(c+h) + (c-h)]$, sehingga

$$f(c) = f\left(\frac{1}{2}(c+h) + \frac{1}{2}(c-h)\right) \leq \frac{1}{2}f(c+h) + \frac{1}{2}f(c-h)$$

$$f(c) \leq \frac{1}{2}f(c+h) + \frac{1}{2}f(c-h)$$

$$2f(c) \leq f(c+h) + f(c-h)$$

$$2f(c) - 2f(c) \leq f(c+h) + f(c-h) - 2f(c)$$

Akibatnya, $f(c+h) - 2f(c) + f(c-h) \geq 0$. Karena $h^2 > 0$ untuk tiap $h \neq 0$, sehingga dapat disimpulkan bahwa $f''(c) \geq 0$. Dengan analogi yang sama dapat dibuktikan f konkaf jika turunan keduanya ≤ 0

D. Pemrograman Linear

Keinginan untuk mendapatkan keuntungan dalam melakukan sebuah usaha mendasari berkembangnya ilmu mengenai pemogramanan linear. Setiap perusahaan atau organisasi memiliki keterbatasan atas sumber dayanya, baik dalam hal jumlah bahan baku, peralatan, tenaga kerja, jam kerja maupun modal. Keterbatasan ini menuntut sebuah perencanaan dan strategi yang dapat mengoptimalkan hasil yang ingin dicapai. Salah satu cara yang telah ditemukan untuk tujuan tersebut adalah pemrograman linear (Eddy Herjanto, 2008 : 9).

Pada dasarnya, pemrograman linear merupakan metode matematika untuk memecahkan masalah dalam mengalokasikan sumber daya yang terbatas untuk mencapai suatu tujuan seperti memaksimumkan keuntungan dan meminimumkan biaya. Pemrograman linear dapat diterapkan dalam masalah bisnis, ekonomi, industri, militer, sosial, teknik, dan lain-lain. Model pemrogamaman linier terdiri

atas sebuah fungsi tujuan dan beberapa fungsi batasan/kendala (Siringoringo, 2005 : 43). Fungsi tujuan merupakan suatu persamaan yang berbentuk linear. Fungsi kendala merupakan persediaan sumber-sumber yang langka yang berkaitan dengan fungsi tujuan. Berikut diberikan definisi fungsi linear dan pertidaksamaan linear..

Definisi 2.5 Fungsi Linear (Winston, 2004:52) *Fungsi* $(x_1, x_2,\ldots, x_n)$ *merupakan fungsi linear jika dan hanya jika fungsi* f *dapat dituliskan* $f(x_1, x_2,\ldots ,x_n) = c_1x_1 + c_2x_2 + \ldots + c_nx_n$*, dengan* $c_1, c_2,\ldots ,c_n$ *merupakan konstanta.*

Berikut diberikan contoh fungsi linear untuk memberikan gambaran berdasarkan definisi di atas.

Contoh 2.3

Fungsi berikut merupakan fungsi linear:

$f(x_1, x_2) = 2x_1 + x_2$

$f(x_1, x_2, x_3) = x_1 - x_2 + 4x_3$

Definisi 2.6 Fungsi Pertidaksamaan Linear (Winston, 2004:52) *Untuk sebarang fungsi linear* $f(x_1, x_2,\ldots ,x_n)$ *dan sebarang bilangan* $b \in R$*, pertidaksamaan* $f(x_1, x_2,\ldots ,x_n) \leq b$ *dan* $f(x_1, x_2,\ldots ,x_n) \geq b$ *merupakan fungsi pertidaksamaan linear.*

Berikut diberikan contoh fungsi pertidaksamaan linear untuk memberikan gambaran berdasarkan definisi di atas.

Contoh 2.4

Fungsi berikut merupakan fungsi pertidaksamaan linear:

$2x_1 + 3x_2 \leq 3$

$2x_1 - x_2 + x_3 \geq 3$

Masalah pemrograman linear pada dasarnya memiliki ketentuan-ketentuan berikut ini (Winston, 2004:53)

1. Masalah pemrograman linear berkaitan dengan upaya memaksimumkan (pada umumnya keuntungan) atau meminimumkan (pada umumnya biaya) yang disebut sebagai fungsi tujuan dari pemrograman linear. Fungsi tujuan ini terdiri dari variabel-variabel keputusan.
2. Terdapat kendala-kendala atau keterbatasan, yang membatasi pencapaian tujuan yang dirumuskan dalam pemrograman linear. Kendala-kendala ini dirumuskan dalam fungsi-fungsi kendala yang terdiri dari variabel-variabel keputusan yang menggunakan sumber-sumber daya yang terbatas itu.
3. Ada pembatasan tanda untuk setiap variabel dalam masalah ini. Untuk sembarang pembatasan tanda menentukan x_i harus non negatif ($x_i \geq 0$) atau tidak dibatasi tandanya *(unretsricted in sign).*

Pemrograman linear merupakan salah satu teknik/metode riset operasi yang digunakan untuk menyelesaikan suatu permasalahan dengan memaksimumkan atau meminimumkan suatu bentuk fungsi objektif atau fungsi tujuan dengan kendala-kendala berupa fungsi yang linear, permasalahan tersebut sering disebut sebagai masalah optimisasi (Rao, 2009:119).

Definisi 2.7 Pemrograman Linear (B. Susanta, 1994:6) *Diberikan fungsi* $f(x_1, x_2, \ldots, x_n)$ *merupakan fungsi linear,* x_j *merupakan variabel keputusan ke-j,* c_j *dan* a_{ij} *merupakan konstanta-konstanta yang diketahui,* b_m *merupakan nilai ruas kanan dari persamaan kendala ke-m yang menunjukkan nilai syarat kendala*

tersebut, untuk setiap i = 1,2, … , m *(indeks untuk jumlah variabel kendala) dan* j = 1,2, … , n *(indeks untuk jumlah variable keputusan).*

Secara umum, masalah pemrograman linear dapat dirumuskan menjadi bentuk berikut :

Memaksimumkan / meminimumkan : $f = C^T X$ (2.1)

dengan kendala : $AX(\leq, =, \geq)B$ dan $X \geq 0$ (2.2)

$C^T = [c_1\ c_2\ \cdots\ c_n]$, (2.3)

$$X = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}, \quad (2.4)$$

$$A = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{21} \\ \vdots & \vdots & \cdots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix}, \quad (2.5)$$

$$\text{dan } B = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{bmatrix}, \quad (2.6)$$

Matriks X merupakan matriks satu kolom dari variabel-variabel yang dicari, dan C^T adalah matriks satu baris untuk setiap koefisien ongkos (c_j). Matriks A merupakan matriks koefisien persamaan kendala, dan B adalah matriks satu kolom dari ruas kanan persamaan kendala. (Bronson & Naadimuthu, 1997 : 20).

Jika (2.1) dan (2.2) dituliskan semua dalam bentuk matriks maka akan menjadi :

Memaksimumkan atau meminimumkan $f = [c_1 \quad c_2 \quad \cdots \quad c_n] \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$,

dengan kendala :

$$\begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \cdots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} (\leq, =, \geq) \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{bmatrix}, dan \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \geq 0$$

Jika bentuk perkalian matriks tersebut diuraikan menjadi penjumlahan aljabar akan menjadi :

Memaksimumkan atau meminimumkan

$$f = c_1x_1 + c_2x_2 + \cdots + c_nx_n = \sum_{j=1}^{n} c_jx_j \quad (2.7)$$

dengan kendala :

$$a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n(\leq, =, \geq)b_1 \quad (2.8a)$$

$$a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n(\leq, =, \geq)b_2 \quad (2.8b)$$

$$\vdots$$

$$a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n(\leq, =, \geq)b_m \quad (2.8c)$$

$$x_1, x_2, \cdots, x_n \geq 0 \quad (2.8d)$$

atau jika ditulis ulang, maka bentuk fungsi kendala (2.8a) – (2.8d) menjadi

$$g_i(x_1, x_2, \cdots, x_n) = \sum_{j=1}^{n} a_{ij}x_j \ (\leq, =, \geq)b_i \text{ dengan } i = 1,2, \cdots, m \quad (2.9a)$$

$$x_j \geq 0 \text{ ,dengan } j = 1,2, \cdots, n \quad (2.9b)$$

E. Pemrograman Nonlinear

Banyak kasus dalam penyelesaian masalah optimisasi yang modelnya tidak dapat dinyatakan dalam bentuk linear. Jika tidak dapat dinyatakan dalam bentuk linear maka bentuk yang didapat adalah nonlinear. Fungsi nonlinear yang biasa dijumpai yaitu fungsi yang memuat perkalian dua variabel bebas atau lebih,

fungsi pangkat$-n$ dengan $n \geq 2$, fungsi eksponen, fungsi trigonometri dan fungsi logaritma.

Pemrograman nonlinear merupakan salah satu teknik dari riset operasi untuk menyelesaikan permasalahan optimisasi dengan fungsi tujuan nonlinear dan fungsi kendala berbentuk nonlinear atau linear. (Bazaraa, 2006:1).

Memilih n variabel keputusan $x_1, x_2, \dots, x_n$ dari daerah fisibel yang diberikan untuk mengoptimisasi (maksimum atau minimum) fungsi tujuan yang diberikan. Permasalahan pemrograman nonlinear secara umum dapat didefinisikan sebagai berikut (Bradley, 1976 : 410).

Memaksimumkan/meminimumkan fungsi tujuan, dalam hal ini f merupakan fungsi nonlinear

$$f(x_1, x_2, , \dots, x_n) \tag{2.10}$$

dengan fungsi kendala dapat berbentuk linear atau nonlinear

$$g_1(x_1, x_2, \dots, x_n)(\leq, =, \geq) b_1 \tag{2.11a}$$

$$\vdots$$

$$g_m(x_1, x_2, \dots, x_n)(\leq, =, \geq) b_m \tag{2.11b}$$

dan batasan nonnegatif pada variabel dengan menambahkan kendala

$$x_1 \geq 0, x_2 \geq 0, \dots, x_n \geq 0 \tag{2.11c}$$

$f(x)$ merupakan fungsi tujuan dan $g_i(x)$ merupakan fungsi kendala dengan b_i menunjukkan nilai syarat kendala tersebut. $f(x)$ dan $g_i(x)$ merupakan fungsi yang kontinu dan *differensiable.*

Persamaan (2.10)−(2.11c) dapat dituliskan dalam bentuk masalah optimisasi yang lebih sederhana sebagai berikut:

Mak/Min $f(x_1, x_2, , \dots, x_n)$ (2.12)

dengan kendala

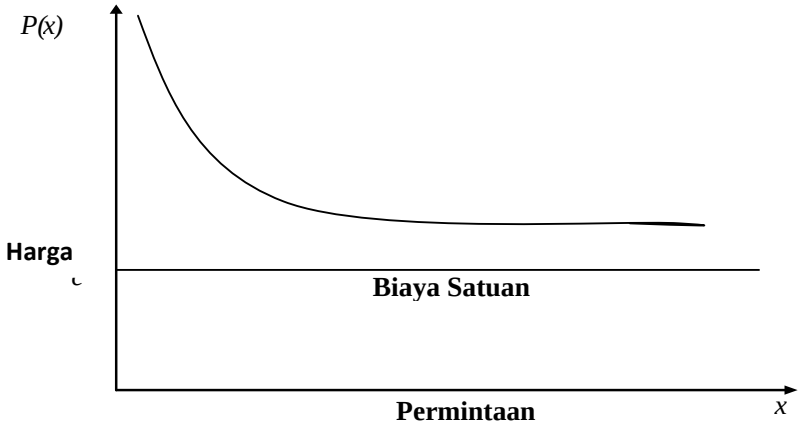
$g_i(x)(\leq, =, \geq)b_i, i = 1,2, \dots, m$ (2.13a)

$x_j \geq 0, j = 1,2,3, \dots, n$ (2.13b)

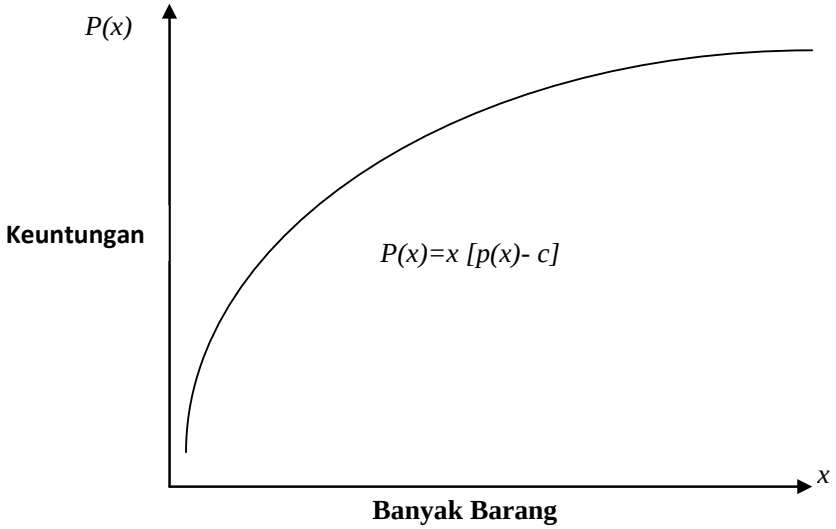
Beberapa kasus yang tidak dapat dimodelkan dalam pemrograman linear karena terdapat beberapa faktor yang menyebabkan ketidaklinearan dalam fungsi tujuan. Sebagai contoh dalam suatu perusahaan besar yang kemungkinan menghadapi elastisitas harga, banyak barang yang dijual berbanding terbalik dengan harganya. Artinya semakin sedikit produk yang dihasilkan maka semakin mahal harganya. Jadi kurva harga permintaan akan terlihat seperti kurva dalam Gambar 2.7, dengan $p(x)$ adalah harga yang ditetapkan agar terjual x satuan barang. Jika biaya satuan untuk memproduksi barang tersebut adalah konstan yaitu di c, maka keuntungan perusahaan tersbut dalam memproduksi dan menjual x satuan barang akan dinyatakan oleh fungsi nonlinear berikut (Hillier , 2001: 655).

$$P(x) = xp(x) - cx \quad (2.14)$$

Gambar 2.8 terlihat misalkan setiap produk dari x jenis produknya mempunyai fungsi keuntungan yang serupa, didefinisikan $P_j(x_j)$ untuk produksi dan penjualan x_j satuan dari produk j dimana $j = 1, 2, \dots, n$, maka secara lengkap fungsi tujuannya yaitu $f(x) = \sum_{j=1}^{n} P_j(x_j)$merupakan penjumlahan dari beberapa fungsi nonlinear.



Gambar 2.7 Kurva Harga Permintaan



Gambar 2.8 Fungsi Keuntungan

Alasan lain yang menyebabkan sifat ketidaklinearan muncul pada fungsi tujuan, disebabkan oleh kenyataan bahwa biaya marginal untuk memproduksi satu satuan barang tergantung pada tingkat produksi. Sebagai contoh, biaya marginal akan turun apabila tingkat produksi naik, sebagai akibat efek dari kurva belajar (*learning curve*). Di lain pihak, biaya marginal dapat saja naik karena dalam ukuran tertentu, seperti fasilitas lembur atau harga barang mahal, sehingga perlu menaikkan produksi.

Sifat ketidaklinearan dapat juga muncul pada fungsi kendala $g_i(x)$dengan cara yang sama. Sebagai contoh, apabila terdapat kendala anggaran dalam biaya produksi total, maka fungsi biaya akan menjadi nonlinear jika biaya produksi marginal berubah. Kendala $g_i(x)$ akan berbentuk nonlinear apabila terdapat penggunaan yang tidak sebanding antara sumber daya dengan tingkat produksi dari masing-masing produksi.

F. Separable Programming

Permasalahan yang semakin kompleks dalam suatu pemrograman nonlinear mendorong terciptanya berbagai metode penyelesaian. Salah satu metode yang dapat digunakan untuk menyelesaiakan pemrograman nonlinear adalah *separable programming*.

1. Pengertian *Separable Programming*

Separable Programming atau yang sering disebut pemrograman terpisah merupakan salah satu metode dalam penyelesaian pemrograman nonlinear dengan

cara mentransformasikan bentuk fungsi nonlinear menjadi fungsi-fungsi linear yang hanya memuat satu variabel saja.

Suatu fungsi $f(x)$ dapat dikatakan terpisah apabila fungsi tersebut dapat dinyatakan dalam bentuk penjumlahan dari fungsi-fungsi yang hanya memuat satu variabel, didefinisikan sebagai berikut (Bazaraa, 2006:684).

$$f(x) = f(x_1, x_2, \dots, x_n) = f(x_1) + f(x_2) + \cdots + f(x_n) = \sum_{j=1}^{n} f_j(x_j) \quad (2.15)$$

Selanjutnya masalah separable programming ditulis dengan masalah P, dengan persamaan (2.15) dapat dituliskan sebagai berikut

Definisi 2.8 *Masalah P (Bazaraa, 2006:684) Diberikan fungsi f_j merupakan fungsi tujuan berbentuk nonlinear dan g_{ij} merupakan fungsi kendala yang dapat berbentuk linear atau nonlinear dengan b_i menunjukkan nilai syarat kendala tersebut, dalam hal ini x_j merupakan variabel bebas. Masalah P didefinisikan sebagai berikut*

Memaksimumkan/meminimumkan

$$Z = \sum_{j=1}^{n} f_j(x_j) \quad (2.16)$$

dengan kendala

$$\sum_{j=1}^{n} g_{ij}(x_j)\ (\leq, =, \geq) b_j, i = 1, 2, \dots, m \quad (2.17a)$$

$$x_j \geq 0; (j = 1,2, \dots, n) \quad (2.17b)$$

Fungsi pada persamaan (2.16) dan (2.17) dapat dijabarkan menjadi

$$Z = f_1(x_1) + f_2(x_2) + \cdots + f_n(x_n) \quad (2.18)$$

$$g_{11}(x_1) + g_{12}(x_2) + \cdots + g_{1n}(x_n)(\leq, =, \geq) b_1 \quad (2.19a)$$

$$g_{21}(x_1) + g_{22}(x_2) + \cdots + g_{2n}(x_n)(\leq, =, \geq)b_2 \quad (2.19b)$$

$$g_{m1}(x_1) + g_{m2}(x_2) + \cdots + g_{mn}(x_n)(\leq, =, \geq)b_m \quad (2.19c)$$

dan $x_1, x_2, \ldots, x_n \geq 0$ (2.19d)

Persamaan (2.18) dan (2.19a)-(2.19d) adalah persamaan fungsi tujuan dan fungsi kendala yang berbentuk penjumlahan dari fungsi-fungsi satu variabel yang disebut masalah *separable programming.*

Contoh 2.5

Diberikan pemrograman nonlinear,

Memaksimumkan $Z = 8x_1 + 14x_2 - 3{x_1}^2 + 9{x_2}^2$

dengan kendala

$${x_1}^2 + 2{x_2}^2 \leq 100$$

$$x_1 + x_2 \leq 10$$

$$x_1,\ x_2 \geq 0.$$

Diperoleh masalah *separable programming* dari fungsi tujuan dan kendala pada Contoh 2.5 sebagai berikut

$$f_1(x_1) = 8x_1 - 3{x_1}^2$$

$$f_2(x_2) = 14x_2 + 9{x_2}^2$$

$g_{11}(x_1) = {x_1}^2,\ g_{12}(x_2) = 2{x_2}^2, g_{21}(x_1) = x_1$, dan $g_{22}(x_2) = x_2$.

2. Hampiran Fungsi Linear Sepotong-sepotong

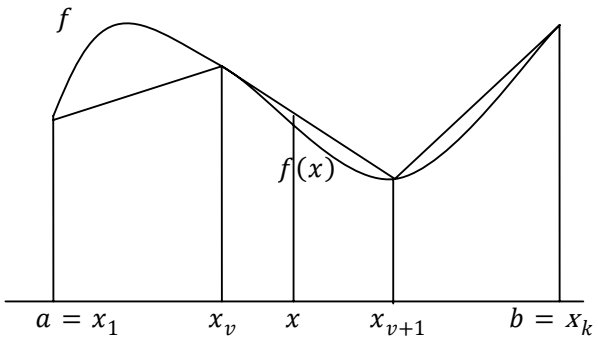
Penyelesaian dalam masalah *separable programming* dapat dilakukan dengan menggunakan beberapa metode diantaranya yaitu metode *cutting plane*, pemrograman dinamik, dan hampiran fungsi linear sepotong-sepotong. Keakuratan dari metode hampiran linear sepotong-sepotong dipengaruhi oleh banyaknya titik kisi. Ada dua cara dalam hampiran fungsi linear sepotong-sepotong, yaitu dengan formulasi lambda (λ) dan formulasi delta (δ) (Bazaraa, 2006:685). Formulasi lambda merupakan formulasi hampiran untuk setiap titik kisi sedangkan formulasi delta merupakan formulasi hampiran untuk setiap interval di antara titik kisi.

Penelitian ini membahas penyelesaian pemrograman nonlinear dengan menggunakan separable programming hampiran fungsi linear sepotong-sepotong formulasi lambda. Sebelum membahas mengenai formulasi lambda terlebih dahulu dibahas mengenai ruas garis

Didefinisikan $f(x)$ merupakan fungsi nonlinear yang kontinu, dengan x pada interval [a,b]. Akan didefinisikan fungsi linear sepotong-sepotong $\hat{f}$ yang merupakan hampiran fungsi f pada interval [a,b]. Selanjutnya interval [a,b] dipartisi menjadi interval-interval yang lebih kecil, dengan titik partisi (*titik kisi*) $a = x_1, x_2, \dots, x_k = b$. Titik-titik kisi tidak harus mempunyai jarak yang sama. Berikut diberikan didefinisikan ruas garis untuk menjelaskan hubungan antara dua titik kisi.

Definisi 2.9 *Ruas Garis (Bazaraa, 2006:684) Diberikan* $x_1, x_2 \in R$. *Himpunan* $S = \{x | x = \lambda x_1 + (1-\lambda)x_2, 0 \leq \lambda \leq 1\}$ *disebut ruas garis yang menghubungkan* x_1 *dan* x_2.

Gambar 2.9 menunjukkan fungsi linear sepotong-sepotong sebagai hampiran fungsi nonlinear f pada interval $[x_v, x_{v+1}]$ dengan sedikit titik kisi.



Gambar 2.9 Fungsi Linear Sepotong-sepotong sebagai Hampiran Fungsi Nonlinear dengan Sedikit Titik Kisi

Misalkan x merupakan titik kisi pada ruas garis yang menghubungkan x_v dengan x_{v+1}, berdasarkan Definisi *2.8.* x dapat dituliskan sebagai berikut

$$x = \lambda(x_v) + (1-\lambda)(x_{v+1}) \text{ untuk } \lambda \in [0,1]. \quad (2.20)$$

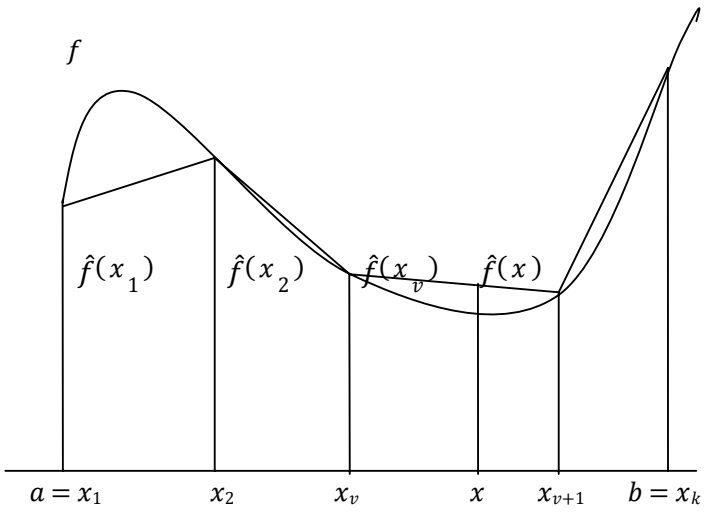
Berdasarkan persamaan (2.20), fungsi $f(x)$ untuk $x \in R$ dapat dihampiri oleh interval $f(x_v)$ dan $f(x_{v+1})$ dengan cara berikut

$$\hat{f}(x) = \lambda f(x_v) + (1-\lambda) f(x_{v+1}) \quad (2.21)$$

Formulasi lambda merupakan hampiran untuk setiap titik kisi dengan menggunakan variabel λ.

Pada Gambar 2.10, untuk sembarang fungsi f didefinisikan pada interval [a,b]. Selanjutnya interval dipartisi menjadi beberapa titik kisi dengan ttik kisi

$a = x_1, x_2, \ldots, x_k = b$. Pada x_1 dihampiri oleh $\hat{f}(x_1)$, x_2 dihampiri $\hat{f}(x_2), x_v$ dihampiri $\hat{f}(x_v)$ dan seterusnya. Titik-titik kisi tidak harus mempunyai jarak yang sama.



Gambar 2.10 Fungsi linear sepotong-sepotong sebagai hampiran fungsi nonlinear dengan formulasi lambda

Secara umum hampiran linear dari fungsi *f(x)* untuk titik-titik kisi $x_1, x_2, \ldots, x_k$ didefinisikan sebagai berikut

$$\hat{f}(x) = \sum_{v=1}^{k} \hat{f}(x_v)\, \lambda_v, \sum_{v=1}^{k} \lambda_v = 1, \lambda_v \geq 0 \qquad (2.22)$$

dengan x yang diperoleh berdasarkan pada interval persamaan (2.20) yaitu

$$x = \sum_{v=1}^{k} x_v\, \lambda_v, \text{ untuk } v = 1, 2, \ldots, k \qquad (2.23)$$

dan terdapat paling sedikit satu λ_v tidak nol atau paling banyak dua λ_v, λ_{v+1} tidak nol dan berdampingan.

Secara umum, dalam setiap dua titik kisi diperoleh satu hampiran sehingga total dari semua hampiran tersebut merupakan hampiran untuk fungsi nonlinear tersebut. Masalah pengoptimuman yang menghampiri masalah P dapat dilakukan dengan mengganti fungsi f_i dan g_{ij} yang nonlinear dengan fungsi linear sepotong-sepotong.

Didefinisikan

$L = \{j | f_j dan\ g_{ij}\ adalah\ fungsi\ linear\ untuk\ i = 1, 2, \dots, m\}$.

Didefinisikan titik-titik kisi x_{vj} untuk $v = 1,2, \dots, k$ pada interval $[a_j, b_j]$ dengan $a_j, b_j \geq 0$ untuk setiap $j \notin L$.

Berdasarkan Persamaan (2.22) dengan titik-titik kisi x_{vj} fungsi f_j dan g_{ij} untuk f_i dan $i = 1,2, \dots, m;\ j \notin L$, maka diperoleh hampiran-hampiran linearnya yaitu

$$\widehat{f_j}(x_j) = \sum_{v=1}^{kj} f_j(x_{vj})\, \lambda_{vj}\ untuk\ \ j \notin L \tag{2.24}$$

$$\widehat{g_{ij}}(x_j) = \sum_{v=1}^{kj} g_{ij}(x_{vj})\, \lambda_{vj}\ untuk\ i = 1,2, \dots, m;\ j \notin L \tag{2.25}$$

dengan $\sum_{v=1}^{kj} \lambda_{vj} = 1$ (2.26a)

$$\lambda_{vj} \geq 0\ untuk\ v = 1,2, \dots, k_j;\ j \notin L \tag{2.26b}$$

dengan x_j yang diperoleh berdasarkan pada Persamaan (2.23) yaitu

$$x_j = \sum_{v=1}^{kj} \lambda_{vj}\, (x_{vj}) \tag{2.27}$$

Untuk mempermudah penulisan, hampiran masalah P ditulis dengan masalah AP. Berdasarkan Persamaan 2.24 – Persamaan 2.26, masalah AP dapat didefinisikan sebagai berikut (Bazaraa, 2006:686)

Masalah AP

Memaksimumkan/meminimumkan

$$Z = \sum_{j \notin L} \widehat{f_j}(x_j) \quad (2.28)$$

terhadap kendala

$$\sum_{j \notin L} \widehat{g_{ij}}(x_j) \, (\leq, =, \geq) b_i, (i = 1, 2, \dots, m) \quad (2.29a)$$

$$x_j \geq 0 \text{ untuk } j = 1, 2, \dots, n \text{ dan } j \notin L \quad (2.29b)$$

Perhatikan bahwa fungsi tujuan dan fungsi kendala pada masalah AP adalah fungsi linear sepotong-sepotong.

Berdasarkan Persamaan 2.28 – Persamaan 2.29, masalah AP dapat dibentuk sebagai masalah LAP yang dituliskan sebagai berikut

Masalah LAP

Memaksimumkan/meminimumkan

$$Z = \sum_{j \notin L} \sum_{v=1}^{kj} f_j(x_{vj}) \, \lambda_{vj} \quad (2.30)$$

Terhadap kendala

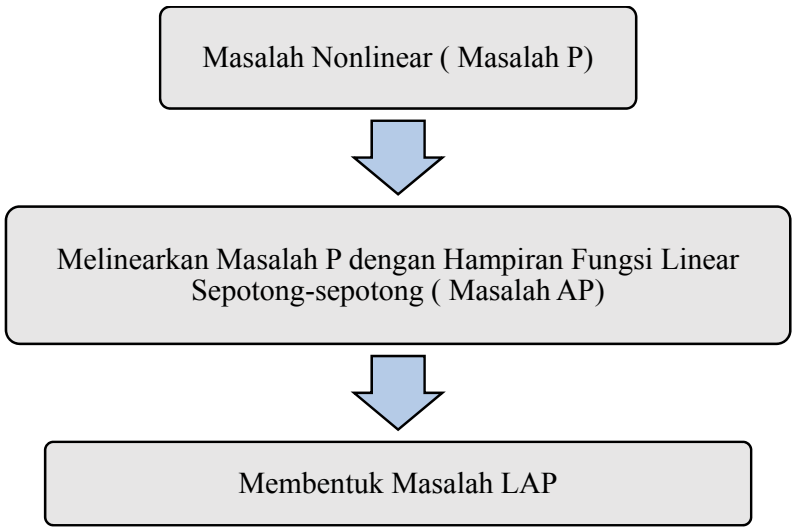
$$\sum_{j \notin L} \sum_{v=1}^{kj} g_{ij}(x_{vj}) \, \lambda_{vj} \, (\leq, =, \geq) b_i, (i = 1, 2, \dots, m) \quad (2.31a)$$

$$\sum_{v=1}^{kj} \lambda_{vj} = 1 \quad (2.31b)$$

$$\lambda_{vj} \geq 0 \; untuk \; v = 1,2, \dots, k_j; \; j \notin L \quad (2.31c)$$

dan terdapat paling sedikit satu λ_{vj} tidak nol atau paling banyak dua $\lambda_{vj}, \lambda_{(v+1)j}$ tidak nol dan berdampingan. Fungsi tujuan dan kendala linear dari persamaan (2.30-2.31) disebut sebagai masalah LAP.

Secara umum *flowchart* proses *separable programming* yang telah dibahas diatas adalah sebagai berikut :



Gambar 2.11 *Flowchart* Proses *Separable Programming*

Selanjutnya masalah LAP dapat diselesaikan dengan metode simpleks biasa. Penelitian ini dalam penyelesaian pemrograman linear menggunakan algoritma genetika dengan bantuan *Software Matlab*.

Setelah mendapatkan penyelesaian optimal dengan algoritma genetika pada masalah minimasi dalam bentuk *separable programming* harus memenuhi syarat bahwa $f_j(x_j)$ harus cembung dan setiap $g_{ij}(x_j)$ adalah cembung (Winston,2004 :714).

Pada penyelesaian *separable programming* berlaku sebagai berikut

Teorema 2.2 (Bazaraa, 2006:689) *Jika* $x_j = \sum_{v=1}^{k} \lambda_{vj}(x_{vj})$ *untuk* $j \notin L$ *merupakan penyelesaian layak pada Persamaan 2.30 – Persamaan 2.31, maka* x_j , $j = 1,2,3, \dots n$ *juga merupakan penyelesaian layak pada Persamaan 2.18 – Persamaan 2.19.*

Bukti:

Berdasarkan Definisi 2.3, karena g_{ij} cembung dengan $j \notin L$ untuk setiap $i = 1,2,3, \dots m$ dan untuk x_{vj} dengan $j \notin L, v = 1,2,3, \dots, k$ diperoleh

$$g_i(x_j) = \sum_{j \in L} g_{ij}(x_j) + \sum_{j \notin L} g_{ij}\,(x_j)$$

$$= \sum_{j \in L} g_{ij}(x_j) + \sum_{j \notin L} g_{ij}\left(\sum_{v=1}^{k} \lambda_{vj}(x_{vj})\right)$$

$$= \sum_{j \in L} g_{ij}(x_j) + \sum_{j \notin L} g_{ij}\,((\lambda_{1j}x_{1j} + \lambda_{2j}x_{2j} + \lambda_{3j}x_{3j} + \cdots + \lambda_{kj}x_{kj})$$

$$\leq \sum_{j \in L} g_{ij}(x_j) + \sum_{j \notin L} \lambda_{1j}g_{ij}(x_{1j}) + \lambda_{2j}g_{ij}(x_{2j}) + \lambda_{3j}g_{ij}(x_{3j}) + \cdots + \lambda_{kj}g_{ij}(x_{kj})$$

$$= \sum_{j \in L} g_{ij}(x_j) + \sum_{j \notin L} \sum_{v=1}^{k} \lambda_{vj}g_{ij}(x_{vj}) \leq b_i.$$

Untuk $i = 1,2,3, \dots m$, selanjutnya $x_j \geq 0$ untuk $j \in L$ dan $x_j = \sum_{v=1}^{k} \lambda_{vj}x_{vj} \geq 0$ untuk $j \notin L$, karena $\lambda_{vj}, x_{vj} \geq 0; v = 1,2,3, \dots k; j \notin L$. Jadi terbukti x_j merupakan penyelesaian yang layak pada Persamaan (2.18)-(2.19).

G. Algoritma Genetika

Algoritma genetika (AG) pertama kali dikenalkan oleh John Holland dari Universitas Michigan pada tahun 1960-an. Kemunculan AG diinspirasikan oleh proses biologi dari teori evolusi Darwin, sehingga banyak istilah dan konsep biologi yang digunakan dalam AG (Chambers, 2000 : 13). AG banyak digunakan untuk memecahkan masalah optimisasi, walaupun pada kenyataannya juga memiliki kemampuan yang baik untuk masalah- masalah selain optimisasi. John Holland menyatakan bahwa setiap masalah yang berbentuk adaptasi (alami maupun buatan) dapat diformulasikan dalam terminologi genetika. Algoritma genetika adalah simulasi dari proses evolusi Darwin dan operasi genetika atas kromosom.

Pada algoritma genetika, teknik pencarian dilakukan sekaligus atas sejumlah solusi yang mungkin, dikenal dengan istilah populasi. Individu yang terdapat dalam satu populasi disebut dengan istilah kromosom. Kromosom ini merupakan suatu solusi yang masih berbentuk simbol. Populasi awal dibangun secara acak, sedangkan populasi berikutnya merupakan hasil evolusi kromosom-kromosom melalui iterasi yang disebut dengan generasi.

Pada setiap generasi, kromosom akan melalui proses evaluasi dengan menggunakan alat ukur yang disebut dengan fungsi *fitness* (kebugaran). Nilai *fitness* dari suatu kromosom akan menunjukkan kualitas dari kromosom dalam populasi tersebut (Zainudin Zukhri, 2014 : 23). Generasi berikutnya dikenal dengan istilah anak (*offspring*) terbentuk dari gabungan dua kromosom generasi sekarang yang bertindak sebagai induk (*parent*) dengan menggunakan operator

penyilangan (*crossover*). Selain operator penyilangan, suatu kromosom dapat juga dimodifikasi dengan menggunakan operator mutasi.

Pemrosesan kromosom-kromosom sebagai sebuah populasi oleh operator genetika terjadi secara berulang (Cole, 1998 : 19). Populasi generasi yang baru dibentuk dengan cara menyeleksi nilai *fitness* dari kromosom induk (*parent*) dan nilai *fitness* dari kromosom anak (*offspring*), serta menolak kromosom-kromosom yang lainnya sehingga ukuran populasi (jumlah kromosom dalam suatu populasi) konstan. Setelah melalui beberapa generasi, maka algoritma ini akan konvergen ke kromosom terbaik.

Menurut Gen dan Cheng Ada tiga kelebihan dari Algoritma Genetika dalam proses pencarian nilai optimal (Zainudin Zukhri, 2014 : 11), yaitu: (a) Algoritma Genetika hanya memerlukan sedikit perhitungan matematis yang berhubungan dengan masalah yang ingin diselesaikan; (b) Operasi evolusi dari Algoritma Genetika sangat efektif untuk mengobservasi posisi global secara acak; dan (c) Algoritma Genetika mempunyai fleksibilitas untuk diimplementasikan secara efisien pada problematika tertentu.

Algoritma genetika sangat tepat jika digunakan untuk menyelesaikan masalah optimisasi yang kompleks dan sukar diselesaikan dengan menggunakan metode konvensional (Supriyanto, 2010 : 4). Sebagaimana halnya dengan proses evolusi di alam, suatu algoritma genetika yang sederhana umumnya terdiri dari tiga operasi, yaitu: operasi seleksi, operasi *crossover* (persilangan), dan operasi mutasi.

Struktur umum dari suatu algoritma genetika terdiri dari langkah-langkah:

a. Membangkitkan Populasi Awal

Proses pembangkitan populasi awal diawali dari pengkodean gen dari kromosom. Satu gen biasanya merepresentasikan satu variabel. Gen dapat diwakili dalam bentuk : bilangan *real*, bit, daftar aturan, elemen permutasi, elemen program, atau representasi lainya yang dapat diimplementasikan untuk operator genetika. Teknik pengkodean ini tergantung pada pemecahan masalah yang dihadapi. Misalnya, pengkodean secara langsung bilangan *real* atau *integer*. Selanjutnya untuk mendapatkan populasi awal metode yang biasa digunakan adalah pembangkitan secara acak.

b. Seleksi

Seleksi digunakan untuk memilih individu-individu mana saja yang akan dipilih untuk proses *crossover* dan mutasi. Selain itu, untuk mendapatkan calon induk yang baik. "Induk yang baik akan menghasilkan keturunan yang baik". Langkah yang dilakukan dalam seleksi ini adalah pencarian nilai *fitnes*s. Nilai *fitnes*s ini nantinya akan digunakan pada tahap-tahap seleksi berikutnya.Untuk itu dapat digunakan rumus

$$fitness = \frac{1}{1+nilai\ fungsi\ objektif} \qquad (2.32)$$

fungsi objektif perlu ditambah 1 untuk menghindari kesalahan yang diakibatkan pembagian oleh 0. Semakin tinggi nilai *fitness* suatu individu semakin besar kemungkinannya untuk dipilih.

Masing-masing individu dalam wadah seleksi akan menerima probabiltas yang tergantung pada nilai *fitnes*snya. Selanjutya akan dicari nilai probabilitasnya dengan rumus

$$P(i) = \frac{fitness(i)}{total\ fitness} \quad (2.33)$$

Setelah didapatkan nilai probabilitasnya selanjutnya dihitung komulatif probabilitasnya. Proses seleksi menggunakan *roulete-wheel* dilakukan setelah didapatkan nilai komulatif probabilitas. Prosesnya adalah dengan membangkitkan bilangan acak R dalam *range* 0-1. Jika $R(k) < C(1)$ maka pilih kromosom 1 sebagai induk, jika $C(k-1) < R < C(k)$ maka pilih kromosom ke-*k* sebagai induk.

c. Crossover

Pindah silang (*Crossover*) adalah operator dari algoritma genetika yang melibatkan dua induk untuk membentuk kromosom baru. *Crossover* menghasilkan titik baru dalam ruang pencarian yang siap diuji. Operasi ini tidak selalu dilakukan pada semua individu yang ada. Jumlah kromosom yang mengalami *crossover* dalam satu populasi ditentukan oleh parameter *crossover probability* (ρ_c). Individu dipilih secara acak untuk dilakukan penyilangan dengan ρ_c antara 0,6 sampai 0,95 (Achmad Basuki, 2003 : 24). Jika *crossover* tidak dilakukan, maka nilai dari induk akan diturunkan kepada anak (keturunan).

Prinsip dari *crossover* adalah melakukan operasi genetika (pertukaran, aritmatika) pada gen-gen yang bersesuaian dari dua induk untuk menghasilkan individu baru. Para *crossover* dilakukan pada setiap individu dengan probabilitas *crossover* yang telah ditentukan.

d. Mutasi

Proses ini berperan untuk menggantikan gen yang hilang dari populasi akibat proses seleksi yang memungkinkan munculnya kembali gen yang tidak muncul pada inisialisasi populasi. Jumlah kromosom yang mengalami mutasi dalam satu populasi ditentukan oleh parameter *mutation probability* (ρ_m). Pada umumnya nilainya adalah $\frac{1}{n}$ (Supriyanto, 2010:10). Proses mutasi dilakukan dengan cara mengganti satu gen yang terpilih secara acak dengan suatu nilai baru yang didapat secara acak. Pertama hitung dahulu panjang total gen yang ada dalam satu populasi.

$$panjang\ total\ gen = jumlah\ gen \times jumlah\ populasi \qquad (2.34)$$

Jika peluang mutasi terlalu kecil, banyak gen yang mungkin berguna tidak pernah dievaluasi. Tetapi jika peluang mutasi terlalu besar, maka akan terlalu banyak gangguan acak, sehingga anak akan kehilangan kemiripan dari induknya dan algoritme kehilangan kemampuan untuk belajar dan melakukan pencarian.

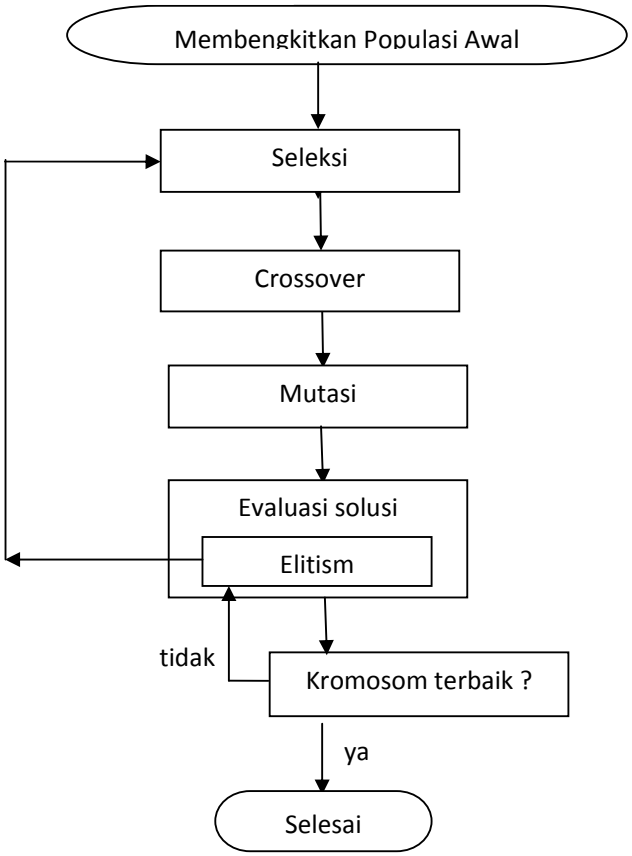
e. Evaluasi Solusi

Evaluasi solusi akan mengevaluasi setiap populasi dengan menghitung nilai *fitness* dari setiap kromosom hingga kriteria berhenti terpenuhi. Namun karena seleksi dilakukan secara acak maka diperlukan langkah untuk menjaga agar individu bernilai *fitness* terbaik tidak hilang selama proses evolusi. Proses ini dikenal dengan nama elitism. Bila kriteria berhenti belum terpenuhi, maka akan dibentuk lagi generasi baru dengan mengulangi langkah sebelumnya tetapi tetap menyertakan individu yang disimpan dalam proses elitism

sehingga hasil perhitungan dapat konvergen. Beberapa kriteria berhenti menurut Budi Sukmawan (2003 : 25) antara lain :

1) Berhenti pada generasi tertentu.
2) Berhenti setelah dalam beberapa generasi berturut-turut didapatkan nilai *fitness* tertinggi/terendah yang tidak berubah.
3) Berhenti bila dalam n generasi berikutnya tidak diperoleh nilai *fitness* yang lebih tinggi/rendah.

Algoritma Genetika secara umum dapat diilustrasikan pada *flowchart* yang ditunjukkan pada gambar 2.12.



Gambar 2.12 Flowchart Proses Algoritma Genetika

Selanjutnya, diberikan contoh penyelesaian pemrograman linear dengan algoritma genetika**.** Contoh berikut diberikan untuk menggambarkan cara kerja algoritma genetika dalam menyelesaikan sebuah masalah optimisasi.

Berikut diberikan gambaran umum proses algoritma yang berlangsung dalam optimisasi.

Contoh 2.6

Minimumkan fungsi : $Z = a + 2b + 3c - 20$

dengan kendala : $a + b \geq 5$,

$b - c \leq -6,$

$0 \leq a, b, c \leq 20$

Penyelesaian :

a. Membangkitkan populasi awal

Karena yang dicari adalah nilai a, b, c, maka variabel a, b, c, dijadikan sebagai gen-gen pembentuk kromosom. Selanjutnya, proses inisialisasi dilakukan dengan cara memberikan nilai awal gen-gen dengan nilai acak sesuai batasan yang telah ditentukan. Misalkan kita tentukan jumlah populasi adalah 6, maka:

Kromosom[1] = $[a; b; c]$ = [06;20;17]

Kromosom [2] = $[a; b; c]$= [06;01;11]

Kromosom [3] = $[a; b; c]$= [05;06;18]

Kromosom [4] = $[a; b; c]$= [08;10;18]

Kromosom [5] = $[a; b; c]$= [13;18;19]

Kromosom [6] = $[a; b; c]$= [11;03;01]

b. Seleksi

Permasalahan yang ingin diselesaikan adalah nilai variabel a, b dan c yang meminimumkan $a + 2b + 3c - 20$, maka fungsi objektif yang dapat digunakan untuk mendapatkan solusi adalah fungsi objektif (kromosom) = $a + 2b + 3c - 20$.

Hitung nilai fungsi objektif dari kromosom yang telah dibangkitkan, yaitu:

f_{ob}(kromosom 1) = 77

f_{ob}(kromosom 2) = 21

f_{ob}(kromosom 3) = 51

f_{ob}(kromosom 4) = 62

f_{ob}(kromosom 5) = 86

f_{ob}(kromosom 6) = 0

Rata- rata dari fungsi objektif adalah : 46

Proses seleksi dilakukan dengan cara membuat kromosom yang mempunyai nilai fungsi objektif kecil mempunyai kemungkinan terpilih yang besar. Untuk itu dapat digunakan fungsi *fitness* sesuai dengan persamaan (2.32)

Fitness 1 = 0,013

Fitness 2 = 0,045

Fitness 3 = 0,019

Fitness 4 = 0,016

Fitness 5 = 0,011

Fitness 6 = 1

Total *fitness* = 1,104

Selanjutnya akan dicari probabilitas terpilihnya menggunakan persamaan (2.33)**.**

P(1) = 0,013/1,104 = 0,012

P(2) = 0,045/1,104 = 0,041

P(3) = 0,019/1,104 = 0,017

P(4) = 0,016/1,104 = 0,014

P(5) = 0,011/1,104 = 0,010

P(6) = 1/1,104 = 0,906

Berdasarkan probabilitas diatas didapatkan kromosom ke 6 yang mempunyai *fitness* paling besar maka kromosom tersebut mempunyai probabilitas untuk terpilih pada generasi selanjutnya lebih besar dari kromosom lainnya. Untuk proses seleksi digunakan *roulete wheel*, untuk itu dicari dahulu nilai kumulatif probabilitasnya:

C(1) = 0,012

C(2) = 0,053

C(3) = 0,070

C(4) = 0,084

C(5) = 0,094

C(6) = 1

Setelah dihitung komulatif probabilitasnya maka proses seleksi menggunakan *roulette wheel* dapat dilakukan. Putar *roulete wheel* sebanyak

jumlah populasi yaitu 6 kali (bangkitkan bilangan acak R sebanyak 6) dan pada tiap putaran pilih satu kromosom untuk populasi baru. Misal didapatkan bilangan acak :

R(1) = 0,195

R(2) = 0,449

R(3) = 0,075

R(4) = 0,161

R(5) = 0,705

R(6) = 0,225

Bilangan acak pertama R(1) lebih besar dari C(5) dan lebih kecil daripada C(6) maka pilih kromosom (6) sebagai kromosom pada populasi baru, dari bilangan acak yang telah dibangkitkan diatas maka populasi kromosom baru hasil proses seleksi adalah:

Kromosom (1) = kromosom(6)

Kromosom (2) = kromosom(6)

Kromosom (2) = kromosom(4)

Kromosom (2) = kromosom(6)

Kromosom (2) = kromosom(6)

Kromosom (2) = kromosom(6)

Kromosom baru hasil proses seleksi adalah :

Kromosom (1) = [11;03;01]

Kromosom (2) = [11;03;01]

Kromosom (3) = [08;10;18]

Kromosom (4) = [11;03;01]

Kromosom (5) = [11;03;01]

Kromosom (6) = [11;03;01]

c. Crossover

Setelah proses seleksi maka proses selanjutnya adalah proses *crossover*. Metode yang digunakan salah satunya adalah one-cut point, yaitu memilih secara acak satu posisi dalam kromosom induk kemudian saling menukar gen. kromosom yang dijadikan induk dipilih secara acak dan jumlah kromosom yang mengalami *crossover* dipengaruhi oleh parameter *crossover probability* (ρ_c). Pada umumnya ρ_c ditentukan mendekati 1, misalnya 0,8 (Suyanto, 2005).

Misal ditentukan *crossover probability* adalah sebesar 0,8. Prosesnya adalah sebagai berikut:

Pertama kita bangkitkan bilangan acak R sebanyak jumlah populasi

R(1) = 0,399

R(2) = 0,187

R(3) = 0,82

R(4) = 0,322

R(5) = 0,161

R(6) = 0,921

Kromosom ke-*k* akan dipilih sebagai induk jika *R(k)* < ρ_c, dari bilangan acak yang telah dibangkitkan maka yang menjadi induk adalah kromosom(1), kromosom(2), kromosom(4) dan kromosom(5).

Posisi *cut-point crossover* dipilih menggunakan bilangan acak 1 – (panjang kromosom-1), dalam persoalan ini berarti dipilih bilangan acak 1-2 sebanyak jumlah *crossover* yang terjadi.

Karena kromosom yang akan mengalami *crossover*, yaitu kromosom(1), kromosom(2), kromosom(4) dan kromosom(5) semuanya memiliki gen yang sama maka hasil persilangannya juga akan sama (tidak ada perubahan).

Dengan demikian populasi kromosom setelah mengalami *crossover* adalah :

Kromosom (1) = [11;03;01]

Kromosom (2) = [11;03;01]

Kromosom (3) = [08;10;18]

Kromosom (4) = [11;03;01]

Kromosom (5) = [11;03;01]

Kromosom (6) = [11;03;01]

d. Mutasi

Jumlah kromosom yang mengalami mutasi dalam satu populasi ditentukan oleh parameter *mutation probability* (ρ_m). Pertama hitung dahulu panjang total gen yang ada dalam satu populasi. Dalam kasus ini panjang total gen sesuai persamaan (2.36) $= 3 \times 6 = 18$.

Untuk memilih posisi gen yang mengalami mutasi dilakukan dengan cara membangkitkan bilangan integer acak antara 1 sampai total gen, yaitu 1 sampai 18. Jika bilangan acak yang dibangkitkan lebih kecil daripada ρ_m

maka pilih posisi tersebut sebagai sub-kromosom yang mengalami mutasi. Misal ρ_m kita tentukan 0,05 maka kemungkinan ada 5% dari total gen yang mengalami mutasi. Jumlah mutasi $= 0{,}05 \times 18 = 0{,}9$ (dibulatkan menjadi 1).

Misalkan setelah dibangkitkan bilangan acak terpilih posisi gen 4 yang mengalami mutasi. Dengan demikian yang akan mengalami mutasi adalah kromosom ke-2 gen nomor 1. Maka nilai gen pada posisi tersebut diganti dengan bilangan acak 0-20

Misal bilangan acak yang terbangkitkan adalah 6, maka kromosom ke-2 gen nomor 1 diganti dengan bilangan 6.

Setelah proses mutasi maka kita telah menyelesaian satu iterasi dalam algoritma genetika atau disebut satu generasi dan didapatkan :

Kromosom (1) = [11;03;01]

Kromosom (2) = [06;03;01]

Kromosom (3) = [08;10;18]

Kromosom (4) = [11;03;01]

Kromosom (5) = [11;03;01]

Kromosom (6) = [11;03;01]

e. Evaluasi

Selanjutnya akan diperiksa fungsi objektif setelah 1 generasi.

f_{ob}(kromosom 1) = 0

f_{ob}(kromosom 2) = -5

f_{ob}(kromosom 3) = 62

f_{ob}(kromosom 4) = 0

f_{ob}(kromosom 5) = 0

f_{ob}(kromosom 6) = 0

Rata-rata fungsi objektif setelah satu generasi $= \frac{0-5+62+0+0+0}{6} = 9{,}5$.

Dapat dilihat dari hasil perhitungan fungsi objektif diatas bahwa setelah satu generasi, nilai hasil rata-rata fungsi objektif mengalami penurunan dibandingkan hasil fungsi objektif pada saat sebelum mengalami seleksi, *crossover* dan mutasi. Hal ini menunjukkan bahwa kromosom atau solusi yang dihasilkan setelah satu generasi lebih baik dibandingkan generasi sebelumnya.

Selanjutnya generasi baru yang telah terbentuk akan mengalami proses yang sama seperti generasi sebelumnya yaitu proses evaluasi, seleksi, *crossover* dan mutasi yang kemudian akan menghasilkan kromosom-kromosom baru untuk generasi yang selanjutnya. Proses ini akan berulang sampai mendapatkan generasi terbaik atau akan berhenti setelah sejumlah generasi yang telah ditetapkan sebelumnya.

Selanjutnya, dengan menggunakan bantuan Matlab (*script* dapat dilihat di lampiran 1) didapatkan solusi dari permasalahan diatas adalah :

nilai a = 5,0003

b = 0,0000

c = 6,0002

Z = a + 2b + 3c - 20 = 3,0009.

Pada umumnya proses Algoritma Genetika untuk mendapatkan hasil optimal membutuhkan proses pengulangan yang cukup panjang. Oleh karena itu, selanjutnya penyelesian optimisasi dengan Algoritma Genetika akan dilakukan dengan bantuan *Software* Matlab.